Quantum Error Correction Below the Surface Code Threshold: Based on the Results of Google Quantum AI*

Jianshuo Gao
School of Physics, Peking University

(Dated: May 15, 2025)

As is known to us, QEC(Quantum Error Correction) is absolutely important in the NISQ(Noisy intermediate-scale quantum) era, and due to its compatibility with two-dimensional architectures and relatively high error thresholds, surface code is considered as a promising protocol to FTQC(Fault Tolerant Quantum Computation). In this review paper, I demonstrate how the suppression factor, code distance and physical error rate interact to enable scalable fault-tolerant quantum memory. By the way we discuss the important role real-time decoding plays in meeting the stringent demands of future large-scale quantum algorithms. What's more, I conclude the main barriers to FTQC, which in turn explains why Google's work is considered a critical step from theoretical feasibility to practical scalability.

Keywords: Quantum Error Correction, Fault Tolerant Quantum Computation, Surface Code, Google Quantum AI

I. INTRODUCTION TO QEC: USING SURFACE CODE AS EXAMPLE

Quantum error correction (QEC) protects fragile quantum information from decoherence and other noise by encoding a single logical qubit into an entangled state of multiple physical qubits without directly copying the quantum state. The digitisation of quantum errors means it is possible to reuse certain techniques from classical coding theory in quantum error correction[1].

However, there remain a number of complications that prevent the straight-forward translation of classical codes to quantum codes. The first complication is the no-cloning theorem for quantum states, which asserts that it is not possible to construct a unitary operator Uclone which performs the following operation

$$U_{clone}(|\psi\rangle \otimes |0\rangle) \rightarrow |\psi\rangle \otimes |\psi\rangle \tag{1}$$

where $|\psi\rangle$ is the state to be cloned.

The second complication in quantum coding arises from the fact that qubits are susceptible to both bit-flips (X-errors) and phase-flips (Z-errors). Quantum error correction codes must therefore be designed with the ability to detect both error-types simultaneously.

The final complication specific to quantum error correction is the problem of wavefunction collapse. For quantum codes, any measurements of the qubits performed as part of the error correction procedure must be carefully chosen so as not to cause the wavefunction to collapse and erase the encoded information.

A. Quantum Redundancy and stabilizer measurement

The encode stage of the two-qubit code, acting on the general state $|\psi\rangle$, has the following action

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \rightarrow |\psi\rangle_L = \alpha|00\rangle + \beta|11\rangle \tag{2}$$

Note that this doesn't correspond to cloning the state as

$$|\psi\rangle_L = \alpha|00\rangle + \beta|11\rangle \neq |\psi\rangle \otimes |\psi\rangle \tag{3}$$

and the operator Z_1Z_2 is said to stabilize the logic qubit $|\psi\rangle_L$ as it leaves it unchanged

$$Z_1Z_2|\psi\rangle_L = Z_1Z_2(\alpha|00\rangle + \beta|11\rangle) = (+1)|\psi\rangle_L \tag{4}$$

So the syndrome extraction stage of the circuit transforms the quantum state as follows

$$\begin{aligned} &E|\psi\rangle_L|0\rangle_A \rightarrow \\ &\tfrac{1}{2}\left(I_1I_2 + Z_1Z_2\right)E|\psi\rangle_L|0\rangle_A \\ &+\tfrac{1}{2}\left(I_1I_2 - Z_1Z_2\right)E|\psi\rangle_L|1\rangle_A \end{aligned} \tag{5}$$

as shown in the FIG.1.

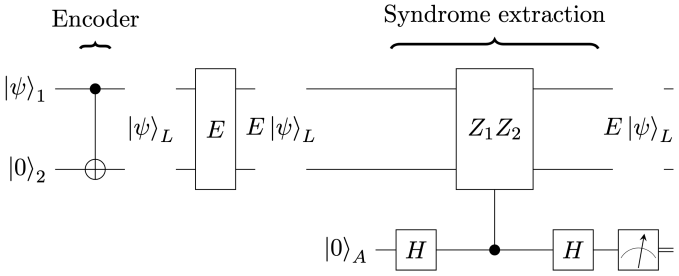

FIG. 1. Encode and syndrome system

There are some simple code such as three-qubit error correction code and Shor code, while I will introduce another type of codes, which is named as stabilizer codes.

* This is a course work paper, nothing new will be included, and there may exist mistakes

B. Stabilizer Codes

Both surface code and color code belong to stabilizer code(due to their speciality, they are also called toric code), and this is such an interesting topic that I would devote a bit more space to explain it.

First we denote any error correction code as $[[n, k, d]]$, where n is the amount of qubits used to encode the logical bits, k means how many bits can this code method encode, and d is the code distance, which I will explain.

The stabilizers P_i of an $[[n, k, d]]$ code must satisfy the following properties:

- They must be Pauli-group elements, $P_i \in G_n$. Here G_n is the Pauli Group over n-qubits.
- They must stabilize all logical states $|\psi\rangle_L$ of the code. This means that each P_i has the action $P_i|\psi\rangle_L = (+1)|\psi\rangle$ for all possible values of $|\psi\rangle_L$.
- All the stabilizers of a code must commute with one another, so that $[P_i, P_j] = 0$ for all i and j. This property is necessary so that the stabilizers can be measured simultaneously (or in a way independent of their ordering).

So the formalism of stabilizers can be defined as this

$$S = \{ P_i \in G_n \mid P_i|\psi\rangle_L = (+1)\,|\psi\rangle, \quad [P_i, P_j] = 0 \} \tag{6}$$

The logical operators satisfy the following properties

- They commute with all the code stabilizers in S.
- They anti-commute with one another, so that $[\bar{X}_i, \bar{Z}_i]_+ = \bar{X}_i\bar{Z}_i + \bar{Z}_i\bar{X}_i = 0$ for all qubits i.

Now I'm going to show a way to calculatek from n. Define the project operator

$$\Pi_C = \frac{1}{2^m}(I + S_1)(I + S_2)\dots(I + S_m) \tag{7}$$

S_i are generators of the stabilizer group. It's easy to see that

$$S_iE|\psi\rangle = ES_i|\psi\rangle, E|\psi\rangle \in CodeSpace$$

and

$$S_iE|\psi\rangle = -ES_i|\psi\rangle, E|\psi\rangle \notin CodeSpace$$

So

$$\Pi_C|\psi\rangle = \begin{cases} |\psi\rangle, & \text{if } |\psi\rangle \in CodeSpace; \\ 0, & \text{if } |\psi\rangle \notin CodeSpace \end{cases} \tag{8}$$

Thus the eigenvalues of Π_C are 1 and 0. Due to this property, we can get $dim(C)$

$$Tr[\Pi_C] = dim(C) \tag{9}$$

Notice the fact that trace of pauli matrix is 0, so there's only identity operator being non-zero, which lives in 2^n Hilbert space

$$dim(C) = \frac{2^n}{2^m} = 2^{n-m} \tag{10}$$

Code space corresponds to logical bits, so the number of bits this can encode is $n-m$, or say this is a $[[n, n-m, d]]$ code, where m is the number of elements in group S.

C. Surface Code

Thanks to Arthur Pesah's blogs, I could have understood surface code very easily.

The surface code can be defined on a square grid of size $L \times L$, where qubits sit on the edges, as shown here for $L = 4$:

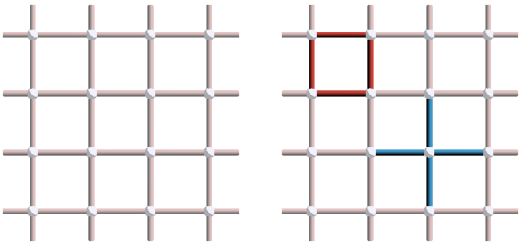

FIG. 2. Lattice on torus and operators

We define two kinds of stabilizers $\bar{X}$(the red plaquettes) and $\bar{Z}$(the blue vertex "+"s)[2][3].

The trick was to draw an edge orthogonal to every blue edge (dashed purple lines on the figure). Formally, this corresponds to representing the operator in the so-called dual lattice, a lattice formed by rotating each edge by 90° (dashed grey lattice in the figure). In this lattice, vertex stabilizers have a square-like shape, similar to the plaquette stabilizers of the primal lattice. Therefore, all the properties we can derive for X stabilizers, errors and logicals can be directly translated to Z operators by simply considering the dual lattice, where Z operators behave exactly like X operators.

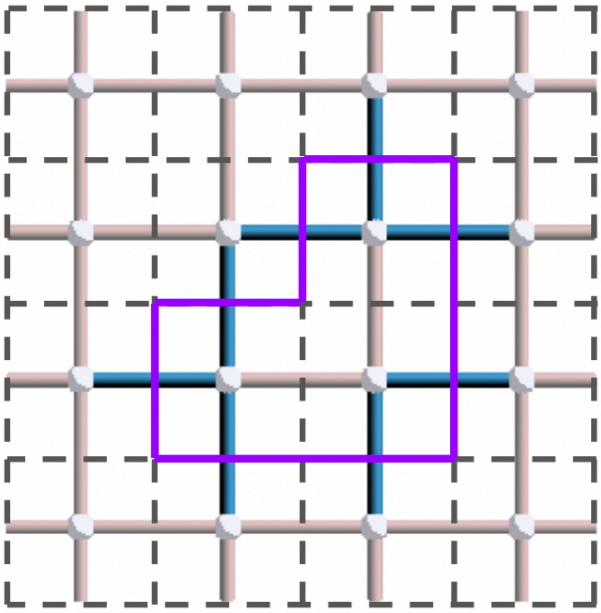

FIG. 3. Dual lattice

In this manner, error excitation only occurs on the boundary of error paths

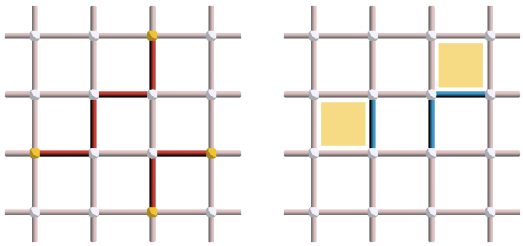

FIG. 4. Error detects

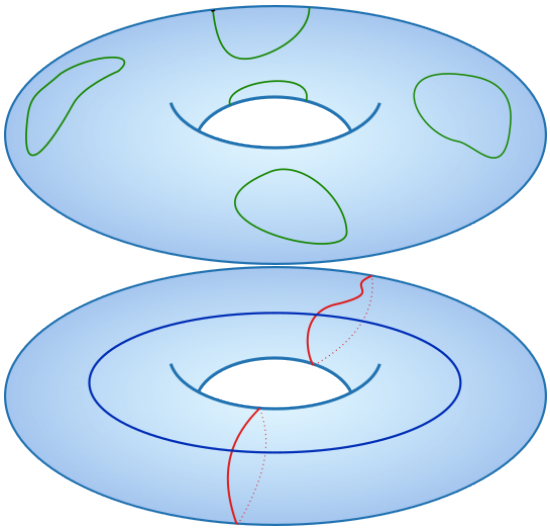

FIG. 5. Operators on the torus

Watch FIG.5, we can find there are two kinds of string operators. We say that two loops on $\mathcal{M}$ are equivalent if there exists a smooth deformation of one loop to the other, meaning that we can smoothly move the first loop to the other loop without cutting it.

Moreover, we say that a loop is contractible, or trivial, if it is equivalent to a point, that is, we can smoothly reduce it until it becomes a single point. All the loops in the figure above are examples of contractible loops on the torus.

While in the figure beneath, One loop (blue) goes around the middle hole, while two loops (red) goes around the hole formed by the inside of the donut. Note that those types of loop (red and blue) are not equivalent to each other, and cannot be deformed to obtain any of the green loops of the first figure neither[4].

Note that in general, the surface code can be defined on any smooth manifold $\mathcal{M}$ by discretizing it. The number of logical qubits of the code is then directly connected to the topological properties of the manifold, and in particular, to the number of holes, or in more technical terms, the first Betti number of the manifold. For instance, for the torus, the fact that $k = 2$ is a consequence of the presence of two holes.

After all we can find that this is a $[[2L^2, 2, L]]$ code.

As for decoding, The goal of it is to find a correction operator that belongs to the same coset as the actual error. Indeed, the product CE of the correction operator with the error is equal to a stabilizer if and only if there is a stabilizer S such that $C = ES$, that is, if C and E belong to the same class. To solve this problem with the information we have, that is, only the syndrome and the error probabilities, the optimal decoding problem, also called maximum-likelihood decoding, can be formulated as finding the coset $\bar{\boldsymbol{C}}$ with the highest probability $\max_{\bar{\boldsymbol{C}}} P(\bar{\boldsymbol{C}})$.

As it happens, this is completely equivalent to solving a famous graph problem, known as minimum-weight perfect matching.

There's also rotated surface code, they are $[[L^2, 1, L]]$, shown in FIG.6. In this manner, qubits are placed on the vertexes.

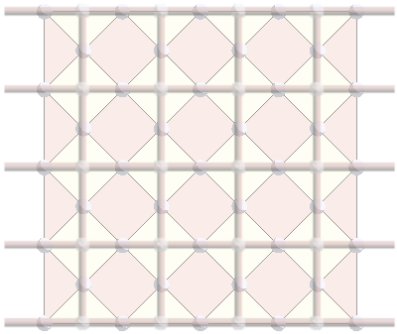

FIG. 6.

That's all we have to know to understand the work of

Google.

II. DEMONSTRATION OF SURFACE CODE BELOW THE THRESHOLD

Quantum error correction is postulated to realize high-fidelity logical qubits by distributing quantum information for many entangled physical qubits to protect against errors. If the physical operations are below a critical noise threshold, the logical error rate should be suppressed exponentially as we increase the number of physical qubits per logical qubit. This behaviour is expressed in the approximate relation[5]

$$\varepsilon_d \propto \left(\frac{p}{p_{\text{thr}}}\right)^{\frac{d+1}{2}} \tag{11}$$

where p is physical error rate, p_{thr} is the threshold error rate of the code, ε_d is logical error rate.

Thus, when $p \ll p_{thr}$, the error rate of the logical qubit is suppressed exponentially in the distance of the code, with the error suppression factor $\Lambda = \frac{\varepsilon_d}{\varepsilon_{d+2}} \approx \frac{p_{thr}}{p}$ representing the reduction in logical error rate when increasing the code distance by two. Although many platforms have demonstrated different features of quantum error correction, no quantum processor has definitively shown below-threshold performance.

This guarantees that with code distance d growing, the probability of logical error happening must decrease. The principle is simple but it is very difficult to realize, and Google first reached this goal as a pioneer.

Along with a high-fidelity processor, fault-tolerant quantum computing also requires a classical co-processor that can decode errors in real time. This is because some logical operations are non-deterministic; they depend on logical measurement outcomes that must be correctly interpreted on the fly. If the decoder cannot process measurements fast enough, an increasing backlog of syndrome information can cause an exponential increase in computation time.

But there's still a lot of barriers in front of us. Although we might, in principle, achieve low logical error rates by scaling up our current processors, it would be resource intensive in practice.[5] Extrapolating the projections and achieving a 10^{-6} error rate would require a distance-27 logical qubit using 1,457 physical qubits, which is difficult to realize in the near future.

Scaling up will bring additional challenges in real-time decoding as the syndrome measurements per cycle increase quadratically with the code distance. On the classical side, we must ensure that software elements including our calibration protocols, real-time decoders and logical compilers can scale to the sizes and complexities needed to run multiple surface code operations.

[1] J. Roffe, Quantum Error Correction: An Introductory Guide, Contemporary Physics **60**, 226 (2019), arXiv:1907.11157 [quant-ph].

[2] A. Y. Kitaev, Fault-tolerant quantum computation by anyons, Annals of Physics **303**, 2 (2003), arXiv:quant-ph/9707021.

[3] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, Topological quantum memory, Journal of Mathematical Physics **43**, 4452 (2002), arXiv:quant-ph/0110143.

[4] A. Córcoles, E. Magesan, S. J. Srinivasan, A. W. Cross, M. Steffen, J. M. Gambetta, and J. M. Chow, Demonstration of a quantum error detection code using a square lattice of four superconducting qubits, Nature Communications **6**, 6979 (2015).

[5] Google Quantum AI and Collaborators, R. Acharya, D. A. Abanin, L. Aghababaie-Beni, I. Aleiner, T. I. Andersen, M. Ansmann, F. Arute, K. Arya, A. Asfaw, N. Astrakhantsev, J. Atalaya, R. Babbush, D. Bacon, B. Ballard, J. C. Bardin, J. Bausch, A. Bengtsson, A. Bilmes, S. Blackwell, S. Boixo, G. Bortoli, A. Bourassa, J. Bovaird, L. Brill, M. Broughton, D. A. Browne, B. Buchea, B. B. Buckley, D. A. Buell, T. Burger, B. Burkett, N. Bushnell, A. Cabrera, J. Campero, H.-S. Chang, Y. Chen, Z. Chen, B. Chiaro, D. Chik, C. Chou, J. Claes, A. Y. Cleland, J. Cogan, R. Collins, P. Conner, W. Courtney, A. L. Crook, B. Curtin, S. Das, A. Davies, L. De Lorenzo, D. M. Debroy, S. Demura, M. Devoret, A. Di Paolo, P. Donohoe, I. Drozdov, A. Dunsworth, C. Earle, T. Edlich, A. Eickbusch, A. M. Elbag, M. Elzouka, C. Erickson, L. Faoro, E. Farhi, V. S. Ferreira, L. F. Burgos, E. Forati, A. G. Fowler, B. Foxen, S. Ganjam, G. Garcia, R. Gasca, É. Genois, W. Giang, C. Gidney, D. Gilboa, R. Gosula, A. G. Dau, D. Graumann, A. Greene, J. A. Gross, S. Habegger, J. Hall, M. C. Hamilton, M. Hansen, M. P. Harrigan, S. D. Harrington, F. J. H. Heras, S. Heslin, P. Heu, O. Higgott, G. Hill, J. Hilton, G. Holland, S. Hong, H.-Y. Huang, A. Huff, W. J. Huggins, L. B. Ioffe, S. V. Isakov, J. Iveland, E. Jeffrey, Z. Jiang, C. Jones, S. Jordan, C. Joshi, P. Juhas, D. Kafri, H. Kang, A. H. Karamlou, K. Kechedzhi, J. Kelly, T. Khaire, T. Khattar, M. Khezri, S. Kim, P. V. Klimov, A. R. Klots, B. Kobrin, P. Kohli, A. N. Korotkov, F. Kostritsa, R. Kothari, B. Kozlovskii, J. M. Kreikebaum, V. D. Kurilovich, N. Lacroix, D. Landhuis, T. Lange-Dei, B. W. Langley, P. Laptev, K.-M. Lau, L. Le Guevel, J. Ledford, J. Lee, K. Lee, Y. D. Lensky, S. Leon, B. J. Lester, W. Y. Li, Y. Li, A. T. Lill, W. Liu, W. P. Livingston, A. Locharla, E. Lucero, D. Lundahl, A. Lunt, S. Madhuk, F. D. Malone, A. Maloney, S. Mandrà, J. Manyika, L. S. Martin, O. Martin, S. Martin, C. Maxfield, J. R. McClean, M. McEwen, S. Meeks, A. Megrant, X. Mi, K. C. Miao, A. Mieszala, R. Molavi, S. Molina, S. Montazeri, A. Morvan, R. Movassagh, W. Mruczkiewicz, O. Naaman, M. Neeley, C. Neill, A. Nersisyan, H. Neven, M. Newman, J. H. Ng, A. Nguyen, M. Nguyen, C.-H. Ni, M. Y. Niu, T. E. O'Brien, W. D. Oliver, A. Opremcak, K. Ottosson,

A. Petukhov, A. Pizzuto, J. Platt, R. Potter, O. Pritchard, L. P. Pryadko, C. Quintana, G. Ramachandran, M. J. Reagor, J. Redding, D. M. Rhodes, G. Roberts, E. Rosenberg, E. Rosenfeld, P. Roushan, N. C. Rubin, N. Saei, D. Sank, K. Sankaragomathi, K. J. Satzinger, H. F. Schurkus, C. Schuster, A. W. Senior, M. J. Shearn, A. Shorter, N. Shutty, V. Shvarts, S. Singh, V. Sivak, J. Skruzny, S. Small, V. Smelyanskiy, W. C. Smith, R. D. Somma, S. Springer, G. Sterling, D. Strain, J. Suchard, A. Szasz, A. Sztein, D. Thor, A. Torres, M. M. Torunbalci, A. Vaishnav, J. Vargas, S. Vdovichev, G. Vidal, B. Villalonga, C. V. Heidweiller, S. Waltman, S. X. Wang, B. Ware, K. Weber, T. Weidel, T. White, K. Wong, B. W. K. Woo, C. Xing, Z. J. Yao, P. Yeh, B. Ying, J. Yoo, N. Yosri, G. Young, A. Zalcman, Y. Zhang, N. Zhu, and N. Zobrist, Quantum error correction below the surface code threshold, Nature **638**, 920 (2025).

[6] Google Quantum AI, R. Acharya, I. Aleiner, R. Allen, T. I. Andersen, M. Ansmann, F. Arute, K. Arya, A. Asfaw, J. Atalaya, R. Babbush, D. Bacon, J. C. Bardin, J. Basso, A. Bengtsson, S. Boixo, G. Bortoli, A. Bourassa, J. Bovaird, L. Brill, M. Broughton, B. B. Buckley, D. A. Buell, T. Burger, B. Burkett, N. Bushnell, Y. Chen, Z. Chen, B. Chiaro, J. Cogan, R. Collins, P. Conner, W. Courtney, A. L. Crook, B. Curtin, D. M. Debroy, A. Del Toro Barba, S. Demura, A. Dunsworth, D. Eppens, C. Erickson, L. Faoro, E. Farhi, R. Fatemi, L. Flores Burgos, E. Forati, A. G. Fowler, B. Foxen, W. Giang, C. Gidney, D. Gilboa, M. Giustina, A. Grajales Dau, J. A. Gross, S. Habegger, M. C. Hamilton, M. P. Harrigan, S. D. Harrington, O. Higgott, J. Hilton, M. Hoffmann, S. Hong, T. Huang, A. Huff, W. J. Huggins, L. B. Ioffe, S. V. Isakov, J. Iveland, E. Jeffrey, Z. Jiang, C. Jones, P. Juhas, D. Kafri, K. Kechedzhi, J. Kelly, T. Khattar, M. Khezri, M. Kieferová, S. Kim, A. Kitaev, P. V. Klimov, A. R. Klots, A. N. Korotkov, F. Kostritsa, J. M. Kreikebaum, D. Landhuis, P. Laptev, K.-M. Lau, L. Laws, J. Lee, K. Lee, B. J. Lester, A. Lill, W. Liu, A. Locharla, E. Lucero, F. D. Malone, J. Marshall, O. Martin, J. R. McClean, T. McCourt, M. McEwen, A. Megrant, B. Meurer Costa, X. Mi, K. C. Miao, M. Mohseni, S. Montazeri, A. Morvan, E. Mount, W. Mruczkiewicz, O. Naaman, M. Neeley, C. Neill, A. Nersisyan, H. Neven, M. Newman, J. H. Ng, A. Nguyen, M. Nguyen, M. Y. Niu, T. E. O'Brien, A. Opremcak, J. Platt, A. Petukhov, R. Potter, L. P. Pryadko, C. Quintana, P. Roushan, N. C. Rubin, N. Saei, D. Sank, K. Sankaragomathi, K. J. Satzinger, H. F. Schurkus, C. Schuster, M. J. Shearn, A. Shorter, V. Shvarts, J. Skruzny, V. Smelyanskiy, W. C. Smith, G. Sterling, D. Strain, M. Szalay, A. Torres, G. Vidal, B. Villalonga, C. Vollgraff Heidweiller, T. White, C. Xing, Z. J. Yao, P. Yeh, J. Yoo, G. Young, A. Zalcman, Y. Zhang, and N. Zhu, Suppressing quantum errors by scaling a surface code logical qubit, Nature **614**, 676 (2023).